Attorney Docket No.: 020706-001200US

PATENT APPLICATION

SYSTEM TO PROVIDE COMPUTING AS A PRODUCT USING DYNAMIC COMPUTING ENVIRONMENTS

Inventor:

Jagadish Bandhole, citizen of India, residing at,
3970 The Woods Drive, #607
San Jose, California 95136

Sekaran Nanja, citizen of the United States, residing at,
5824 Chambertin Drive
San Jose, CA 95118

Shan Balasubramaniam, citizen of India, residing at,
1929 Crisanto Avenue, Apt. 204
Mountain View, CA 94040

Assignee:

Jareva Technologies, Inc.
924 Borregas Avenue
Sunnyvale, CA 94089

Entity: Small

TOWNSEND and TOWNSEND and CREW LLP
Two Embarcadero Center, 8th Floor
San Francisco, California 94111-3834
Tel: 415-576-0200

A PLATFORM TO PROVIDE COMPUTING AS A PRODUCT USING DYNAMIC COMPUTING ENVIRONMENTS

CROSS-REFERENCES TO RELATED APPLICATIONS

[01] This application is a Continuation-in-Part Application of U.S. Patent Application 09/861483. This application is also related to U.S. Patent Application Nos. 09/663252 and 09/662990, respectively entitled "User Interface for Dynamic Computing Environment Using a Allocable Resources" and "System for Configuration of Dynamic Computing Environment Using a Visual Interface," filed September 15, 2000. Both applications and their disclosures are incorporated herein by reference for all purposes.

BACKGROUND OF THE INVENTION

[02] The present invention relates in general to information processing, and more specifically to a system that facilitates dynamic allocation and de-allocation of computing resources to provide a number of virtual computing platforms as a computing product.

[03] Today, computers are increasingly being used in almost every area of commerce, education, entertainment and productivity. With the growing popularity of the Internet, corporate and campus intranets, home networking and other networks, the trend is to use multiple computers, or processing platforms, to perform tasks and provide services. Thus, the use of computers and computing devices have become commonplace in day-to-day activities of large numbers of users from different walks of life, including those with little to no knowledge of how the portability of computer applications depends on specific computing platforms.

[04] A "platform," includes the underlying hardware or software (e.g., operating system, applications, utilities, and other processes) of a computer system. The platform defines a standard around which a computing system can be developed. Once the platform has been defined, software developers can produce appropriate software and users (e.g., consumers) can purchase appropriate hardware and software applications for the platform. Any number, type and combination of hardware and software can comprise a platform, or environment.

[05] For instance, a platform configured to provide a search service on the Web might include a Linux server running Apache web server software, a Solaris server running a custom application server software and Oracle database software, a 100 Mbps Ethernet LAN connecting the servers, and the Internet.

[06] "Computing" in connection with the platform refers to the activity by one or more users interacting with a computing environment, or platform, that includes a combination of hardware, software, and network resources. Such interaction by a user may be in the form of using the environment to accomplish a task using, for example, application software operably compatible with the environment. For instance, the user may interact with a platform to edit a document, send an email, execute a search using a search engine service, or any equivalent application generally known in the art. Computing also includes programming or configuring the computing environment itself to modify the operability of the computing environment.

[07] In traditional models of computing, users acquire the components of the environment, configure them as needed, and maintain them through a period of use. A disadvantage of the traditional models of computing require the users to purchase or lease the components individually to establish a platform for meeting their computing needs. For instance, a word processing user may purchase a personal computer, an operating system, and a word processing application program. The user then typically installs the operating system on the computer and the word processor on the operating system. Similarly, a search service provider will purchase the hardware, such as two server computers, and will then install the software (e.g., the web server, the application server and the operating systems). Thereafter the provider will connect the hardware to the network, connect the network to the Internet, and then configure the software for communication according to a specific communication protocol (e.g., configure the web server to accept requests from clients and to obtain from the application server responses for the said request).

[08] Traditional computing models include running a single application as a stand-alone application on a single computer as well as a "client-server" whereby a server computer on the Internet is used to transfer information to a client computer. Typically, the client computer is located at an end user's location, such as a personal computer in a user's home. This allows large amounts of information to be stored in, and accessed from, the server computer by many client computers. The client computers can access the server computer simultaneously. Another approach allows a user to obtain portions of executable programs from the server to operate an application program in functional "pieces" or components, on the client computer. For example, a user can run a word-processing program in a client-server mode where the server provides only those portions of the word-processing software to the user's computer on an as-needed basis.

[09] Traditional computing models have the drawback of offering relatively limited choices. That is, users of today's computing services generally are required to choose to invest capital in specific computing devices while foregoing other types of devices. Consumers who engage in computing need to be knowledgeable of the underlying platform to ensure a purchased application software will operate correctly.

[10] A few traditional models have attempted to provide computing as service. Although a timesharing model of early mainframe computers was employed to provide a pay-per-use pricing model, the pricing model failed to address other issues. For example, users are not presented with any choices as to how to modify or configure the associated computing environment or how to network additional resources, such as bandwidth or IP Addresses, with a conventional pay-per-use pricing scheme.

[11] A well-known computing model referred as the "Application Service Provider" or ASP model eliminates the acquisition (and maintenance) of a computing infrastructure and introduces pay per use. This model removes the application from the end-user and might employ one or more servers. The ASP model allows a primary server to host a client-server application, or to host any type of data-processing resource such as a database, user interface, program component, data object, etc. The application can appear to the client as being hosted by the primary server when it is actually being provided by one or more other servers. The other servers can provide the application, or components, by having the client directly access the other server, or having the client access the other server through the primary server.

[12] The drawback to this model, as well as other similar known models, includes an inherent inability to generalize and to scale for multiple applications and computing platforms. Another drawback of the ASP model is that it also eliminates user choices; one gets to choose the application, but not other components, such as hardware or network. This is a limitation for users which require high performance and high bandwidth as opposed to other users willing to forego either or both of these requirements for reduced costs. Yet another drawback with ASP models is that the applications are not easily customized for use in an ASP service. Still yet another drawback with ASP models is that often, ASP-ized applications are to be rewritten from scratch. So for new applications, and for in-house applications developed by organizations without core competence in ASP technology, the ASP model is not an effective solution. Still yet another drawback with ASP models is that the ASP model enables the use of the software but does not support programmability.

[13] Thus, it is desirable to provide a system that improves upon the prior art.

SUMMARY OF THE INVENTION

[14] The present invention enables computing to be provided as a packaged product or as a remote resource to users. Computing is delivered as a product or a resource by providing dynamic computing environments to users based on users' choices of virtual components (hardware, software or network components). A customer can choose the components and configure a computing environment. The system packages this environment and makes it available for users to compute. A service provider can use the system to create computing environments, automatically, on demand and thus providing computing as a remote resource to customers. The system monitors the usage of the customers and they are billed accordingly. In either case users can carry out their computing activity remotely using a client device such as a web browser.

[15] In one embodiment the invention provides a system to provide computing as a resource to a user. The system includes a framework for providing a dynamic computing environment using allocable resources; and wherein the dynamic computing environment is used for computing by the user.

BRIEF DESCRIPTION OF THE DRAWINGS

[16] Figure 1A is a first illustration of basic hardware components;

[17] Figure 1B is a second illustration of basic hardware components;

[18] Figure 1C is a third illustration of basic hardware components;

[19] Figure 2 is a block diagram illustrating an architecture of a system for hosting one or more DCE's;

[20] Figure 3 illustrates resources used in a computing environment;

[21] Figure 4 is a flow chart illustrating the steps for a user to purchase computing as a packaged product; and

[22] Figure 5 is a block diagram of a platform for providing computing as a remote resource.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

[23] Detailed descriptions of the embodiments are provided herein. It is to be understood, however, that the present invention may be embodied in various forms. Therefore, specific details disclosed herein are not to be interpreted as limiting, but rather as a basis for the claims and as a representative basis for teaching one skilled in the art to employ the present invention in virtually any appropriately detailed system, structure or manner.

[24] The present invention enables computing resources and the activity of computing to be provided to a user as a packaged product as well as a service. A platform can be any combination of hardware and software components, or other resources. Examples of resources include memory space, processing cycles, network or bus bandwidth, IP addresses, timeslots, etc. Resources further include intangible assets such as the legal right (e.g., license) to use hardware or software. In general, a resource is any asset - including components, services, rights, obligations or other principle or effect - that enables computing.

[25] According to an embodiment of the present invention, a customer can purchase or lease computing resources without acquiring hardware or software components and without managing the requisite infrastructure. The customer does not have to have specialized knowledge of the underlying components of the infrastructure.

[26] This is achieved by a customer specifying a configuration of a computing environment for use, remotely if needed, using a suitable interface. Once specified, a system provides a platform that automatically create this environment dynamically to provide for computing of the present invention by allocating the requisite resources and make them available for the customer, which may limited to a specific requested time period.

[27] According to the present invention, the resources for such a processing network are fully selectable and allocable by a system architect. In a specific embodiment, a primary company, Jareva Technologies, Inc.® provides proprietary technology to a system architect for designing a system by allocating resources and specifying how the resources are to be used. The system architect can be an individual, corporate entity, etc. The system is referred to as an "environment" – or more specifically as a "computing environment" and the primary provider of such an environment is referred to as an Environment Service Provider (ESP). A typical system architect is referred to as the "customer." The primary provider obtains revenue for providing the resources and the tools to easily select, allocate, configure and run the environment.

[28] Dynamic computing environments and the process of creating them automatically are described in detail in related co-pending U.S. Patent Ser. No. 09/663,252; filed September 15, 2000, entitled "USER INTERFACE FOR DYNAMIC COMPUTING ENVIRONMENT USING ALLOCABLE RESOURCES." The interfaces for specifying the configuration of a computing environment and for (remote) access to a computing environment are described in detail in co-pending U.S. Patent Ser. No. 09/662,990; filed September 15, 2000, entitled "SYSTEM FOR CONFIGURATION OF DYNAMIC COMPUTING ENVIRONMENTS USING A VISUAL INTERFACE"

[29] The present invention allows fast, efficient selection and configuration of processing networks, which can then be accessed and managed remotely. The processing network is referred to as a system including "resources." A system resource is any hardware, software or communication components in the system. For example, discrete hardware devices include processing platforms such as computers or processors, mobile/laptop computers, embedded computing devices, hand-held computers, personal digital assistants, point-of-sale terminals, smart-card devices, storage devices, data transmission and routing hardware etc., without limitation. Additionally, computer peripherals such as monitors, input/output devices, disk drives, manufacturing devices, or any device capable of responding to, handling, transferring or interacting with digital data are also resources. Software, or any other form of instruction, is executed by processors in the system and is also a type of resource. Finally, communication resources are also part of the system such as a digital network's hardware including the network's configuration and topology, where control of the network is provided by software and/ or hardware. Additionally, the network may be based on wired connections or wireless connections. For instance, the network hardware and software may be based on Bluetooth wireless standards.

[30] For example, a processing network of a general consumer might include a PDA and a cell phone, each connected by wireless channels to a single personal computer, which in turn is connected to an email server at a remote location through the Internet. As another example, a processing network might include a personal computer running Microsoft Windows 98 operating system, a lap-top computer running Linux operating system, and another personal computer running Windows NT operating system along with router and firewall software, wherein all three computers are connected using a local Ethernet hub, and the router software routes connections to the Internet.

[31] The specific embodiment of the present invention allows fast allocation and configuration of resources such that different environments can be created from the same resources within minutes, or even seconds. This allows "time sharing" of overall resources so that a first environment can be "alive" or operative for a time period defined by the system architect (e.g., daily two-hour slot), followed by second, third and fourth environments being instantly created for the next four hours for three different customers, and so on. After a time period expires, such environments might either manually or automatically de-allocate such resources. Since these "computing environments" can be dynamically configured and re-configured out of the same set of resources, these will also be referred to as "Dynamic Computing Environments".

[32] A specific embodiment allows customers to create a computing environment from a remotely-accessible user interface such as a web page on the Internet. Thus, the customer can create, modify and operate the environment from anywhere in the world. Since the resources, in turn, can communicate over networks, including the Internet, this approach eliminates the cost of shipping hardware and software. Hardware and software designers, programmers, testers or other personnel using an environment according to the present invention can, similarly, be located anywhere in the world such that labor costs are optimized.

[01] The creation of dynamic computing environments ("DCE") is automatic. For example, a customer can request a web-site simulator using twelve web-page servers on a Microsoft® NT platform, two disk arrays at a specific bandwidth and storage capacity, two caching servers and 200 clients running Netscape Navigator™ under Microsoft Windows® 2000 using Pentium III™ processors at under 800 MHz. Such an environment is created and destroyed, and even re-created automatically, without human intervention each time. Unlike the conventional computing infrastructure, according to an embodiment of the present invention there is no need to physically couple or de-couple, each physical machine or resource to each other upon adding or removing such resources. There is no need to set-up Internet Protocol (IP) addresses or other network settings, or install operating systems and associated application programs on one or more physical machines. All such activities on a DCE can be performed automatically without user intervention.

[34] According to the present invention, the DCE is a virtual computing system including a network comprising a number of distinct types of machines and a network connecting them. For example, a system architect might require a DCE to include a Sun Sparc running a certain version of Solaris O/S coupled to a Linux machine. The present invention enables the separation of the activity of designing a DCE, from the activity of actually creating the DCE. Designing a DCE includes choosing the specific hardware, choosing the operating systems or other software, and choosing the specific interconnections, etc. Creating a DCE includes allocating the resources, installing the operating systems and other software, etc. Furthermore, the present invention automates the process of creating the DCE. A DCE for which resources have not been allocated yet will also be referred to as a virtual computing environment. Similarly, a computing device (or a subnet) that is part of a DCE also be referred to as a virtual computing device (or a virtual subnet).

[35] The present invention provides a framework that enables configuring, provisioning, accessing and managing DCEs remotely. Configuring a DCE involves choosing the resources and their interconnections. The present invention supports operations for making such design choices through appropriate programmable interfaces. The interfaces can be used interactively through a graphical user interface such as a web page or non-interactively through a program script. Provisioning a DCE involves allocation of physical resources required for a DCE to function. The present invention manages the physical resources needed for provisioning DCEs and supports operations for allocating/de-allocating these resources. Accessing a DCE involves accessing one or more devices and/or sub-networks within the DCE. The present invention supports operations for accessing the components of a DCE. For instance, when a user needs to copy data from a specific computer to a backup storage device, operations involving "read" access to the computer and its local storage, "write" access to the storage device, and access to the network for transmitting the data will be used by the present invention to meet the user's needs. Managing a DCE involves managing the components of a DCE, such as a personal computer, a network router, etc.

[36] In one embodiment of the present invention, a system provide a framework for administering DCEs is implemented as a distributed system consisting of different software programs running on different computers and networking hardware. Administering DCEs, as described herein refers to the configuring, provisioning, accessing, and managing of dynamic computing environments. In a further embodiment, the present invention permits "virtual" hosting of dynamic computing environments. As used herein, the term "virtual" specifies that neither the requisite devices nor the network need to be physically accessible to users. Further, in accordance with this embodiment, the hosting process may be initiated or terminated by users at will, from any geographic location. Thus the administrative framework allows users to remotely configure, provision, access, and manage DCEs.

[37] A further understanding of embodiments of the present invention will be gained with reference to the diagrams and the descriptions that follow.

[38] Figures 1A, 1B, and 1C illustrate basic hardware components suitable for practicing the present invention. Figure 1A is an illustration of computer system 1 including display 3 having display screen 5. Cabinet 7 houses standard computer components (not shown) such as a disk drive, CDROM drive, display adapter, network card, random access memory (RAM), central processing unit (CPU), and other components, subsystems and devices. User input devices such as mouse 11 having buttons 13, and keyboard 9 are shown. Other user input devices such as a trackball, touch-screen, digitizing tablet, etc. can be used. In general, the computer system is illustrative of but one type of computer system, such as a desktop computer, suitable for use with the present invention. Computers can be configured with

many different hardware components and can be made in many dimensions and styles (e.g., laptop, palmtop, server, workstation, mainframe). Any hardware platform suitable for performing the processing described herein is suitable for use with the present invention.

[39] Figure 1B illustrates subsystems that might typically be found in a computer such as computer 1. In Figure 1B, subsystems within box 20 are directly interfaced to internal bus 22. Such subsystems typically are contained within the computer system such as within cabinet 7 of Figure 1A. Subsystems include input/output (I/O) controller 24, System Memory (or random access memory "RAM") 26, central processing unit CPU 28, Display Adapter 30, Serial Port 40, Fixed Disk 42, Network Interface Adapter 44, which in turn is coupled electrically to a network. The use of bus 22 allows each of the subsystems to transfer data among subsystems and, most importantly, with the CPU, where the CPU might be a Sparc, an Intel CPU, a PowerPC, or the like. External devices can communicate with the CPU or other subsystems via bus 22 by interfacing with a subsystem on the bus. Thus, Monitor 46 connects with Display Adapter 30, a relative pointing device (e.g. a mouse) connects through Serial Port 40. Some devices such as Keyboard 50 can communicate with the CPU by direct means without using the main data bus as, for example, via an interrupt controller and associated registers.

[40] As with the external physical configuration shown in Figure 1A, many subsystem configurations are possible. Figure 1B is illustrative of but one suitable configuration. Subsystems, components or devices other than those shown in Figure 1B can be added. A suitable computer system can be achieved without using all of the subsystems shown in Figure 1B. For example, a standalone computer need not be coupled to a network so Network Interface 44 would not be required. Other subsystems such as a CDROM drive, graphics accelerator, etc. can be included in the configuration without affecting the nature or functionality of the system of the present invention.

[41] Figure 1C is a generalized diagram of a typical network that might be used to practice an embodiment of the present invention. In Figure 1C, network system 80 includes several local networks coupled to the Internet. Although specific network protocols, physical layers, topologies, and other network properties are presented herein, the present invention is suitable for use with any network.

[42] In Figure 1C, computer USER1 is connected to Server1, wherein the connection can be by a network, such as Ethernet, or Asynchronous Transfer Mode, or by a modem, or by other means. The network provides the communication means, such as physical inter-connective links comprising copper wire, fiber optic cable, or the like, for transmitting and

receiving signals. Wireless communication means, such as radio waves or the like, are also understood to provide means to transfer information from a source to a destination. Hence, the communication link need not be a wire but can be infrared, radio wave transmission, etc. Server1 is coupled to the Internet. The Internet is shown symbolically as a collection of server routers 82. Note that the use of the Internet for distribution or communication of information is not strictly necessary to practice the present invention but is merely used to illustrate a specific embodiment, below.

[43] Further, the use of server computers and the designation of server and client machines is not crucial to an implementation of the present invention. USER1 Computer can be connected directly to the Internet. Server1's connection to the Internet is typically by a relatively high bandwidth transmission medium such as a T1 line or T3 line. Similarly, other computers 84 are shown utilizing a local network at a different location from USER1 Computer. The computers at 84 are coupled to the Internet via Server2. USER3 and Server3 represent yet a third installation. In a specific embodiment, a user of the present invention operates a user interface associated with computers 84 to at least virtually configure one or more computing devices as a subnet. Note that the use of the term "computing device" includes any processing device or platform such as a web television device, personal digital assistant (e.g., a Palm Pilot manufactured by Palm, Inc.), cellular telephone, etc.

[44] As is well known in the art of network communications, network is configured to communicate electrical information, such as a computer data signal comprising data (e.g., binary data bits) superimposed upon a radio or any other carrier wave. A person having ordinary skill in the are would appreciate that a carrier wave is electromagnetic energy propagated from a source by radiation, optical or conduction waves and is suitable for embodying an information-bearing signal, such as a computer data signal. In one embodiment, a carrier wave behaves, or is modulated, according to a network protocol, such as or Ethernet, IEEE 1394, TCP/IP, or any other communication protocol, so as to include computer data information. The carrier wave can be, for example, a direct current, an alternating current, or a pulse chain. In modulation of the carrier wave, it may be processed in such a way that its amplitude, frequency, or some other property varies so as to embody data for transfer.

[45] Figure 2 is a block diagram illustrating the N-Tier architectural structure of system 200 for hosting one or more DCEs according to the present invention. In Figure 2, system 200 comprises a number of tiers, namely a switch tier 203, a web tier 205 usability tier 207, middleware tier 209, application logic tier 211 and data center tier 213. One or more of the tiers are implemented using software (proprietary or third-party), or hardware or a combination thereof. Switch tier 203 includes a router 217 for routing data packets through the network, a firewall 218 and a load balancer 219 for balancing the load on web tier 205. The load balancer 219 ensures that each of the web servers in the web tier 205 receives roughly equal amounts of load and if one of the web servers go down (i.e., becomes inoperable) the traffic is routed to other web servers in the web tier 205.

[46] Exemplary load balancer 219 uses IP packets based load-balancing. Of course, one having ordinary skill in the art would appreciate that any other load balancing scheme may be used without affecting the nature of the switch tier 203 or any other tier. Router 217 may be a Cisco 7200 Series™ router available from Cisco, Inc.®, or alternatively, router 217 may be any other suitable type routers, or an equivalent device that provides substantially the same functionality. Web tier 205 comprises one or more web servers such as a Linux box running Apache web server, for example, or other comparable type web servers. Usability tier 207 provides various services including load balancing (for the app-servers), billing, session management, security (SSL), and fault tolerance. SSL refers to Secure Socket Layer, which is a protocol developed by Netscape for transmitting private documents via the Internet.

[01] Middleware tier 209 contains one or more application servers 221, 223 and a module 225 for implementing look up event and services. The primary functionality of the middleware tier is to delegate requests to specific services that are responsible for specific actions. For instance, these actions may involve accessing the database, accessing the storage, or accessing a computing device. The processes running on the application servers 221 and 223 make such delegation decisions and are further illustrated in Figure 3. BEA WebLogic™ servers running on a Solaris® platform, for example, or the like are suitable to implement application servers 221 and 223.

[48] Since one or more of the services in the Application logic tier 211 may be replicated and be running on independent physical machines, they need to be "looked-up" for availability. Using such a lookup service will allow the services in the Application logic tier 211 to be started or shut down asynchronously. That is, the starting or shutting down the services related to logic tier 211 need not be synchronized with the processes on the application servers 221 and 223. For instance, increasing the number of Linux boxes – as capacity devices in Data Center tier 213 – may require increasing the number of Linux device services, and this can be done without the knowledge of the application server(s) by automatically replicating the Linux device services and notifying the lookup service. In addition, the services in the Application logic tier 211 may have to notify events (such as a storage unit is full, or there are no more Linux boxes available) to the application servers 221 and 223. Such notification can be done through Lookup/Event services. Lookup/Event services can be provided through Sun Microsystems' Jini software layer, for example. Of course, other implementations of the lookup/event services using proprietary or third party software are possible.

[49] Application logic tier 211 provides a variety of operating systems device services such as Windows 227, Linux 229, Unix 231 device services. These device services are responsible for managing physical devices available in the data center tier 213. User management service 233 is implemented within application logic tier 211 and establishes and maintains each user's configured virtual machines within a DCE. Such information is stored in the database associated with the application logic tier 211. Data center tier 213 includes various operating system platforms and processors, also selectable by the user. Data center tier 213 also includes networking and storage resources as well. Although not shown, one of ordinary skill in the art will realize that one or more of the aforementioned tiers and components therein can be implemented using third party providers, dedicated custom modules or software and hardware or a combination thereof.

[50] The framework 342 shown in Figure 3 illustrates the typical resources required for a computing environment (such as Capacity devices or CPUs, Storage servers, and Network switches, Licenses), and the infrastructures required for automatically creating a computing environment (such as the application server 311 and its components, the database 345 and its tables). This framework can accept specifications of a computing environment (i.e., how is it to be configured), can create an environment automatically by allocating resources, can enable customer access to the created environment, and can enable management of the environment for continued use. Furthermore this framework allows multiple Dynamic Computing Environments to be created concurrently out of a resource pool.

[51] In one embodiment of the present invention a system using the framework 342 is implemented for providing users with computing as a product. More specifically, in this embodiment a customer can choose specific components required for computing and the system will package the components to provide an environment that can be used for computing by the customer. For instance, the email customer may specify the preferred email software. The system will choose a compatible operating system, a compatible client device running the operating system and the email software, acquire licenses, network connections etc.; the system will then package these resource to provide an email client product to the customer. The customer pays for the product as a whole not separately for the hardware,

software, or for the network connection. Also, the customer need not maintain any of the components and need only learn to access the email system as a whole.

[52] In a further embodiment of the present invention a system using the framework 342 is implemented for providing users with computing a resource. More specifically, in this embodiment a service provider can configure the components required for computing and the system will provide an environment that can be used by one or more customers concurrently. A computing service provider can specify the components of a computing environment and request that multiple copies of the environment be created. The system will create the environments, will allow concurrent access to the environments by different customers, will meter usage for each customer (say in terms of CPU time, storage used, network bandwidth used etc.), will guarantee the quality of service for each customer by monitoring the resources. The customers have secure and isolated access to the environments i.e., first customer's environment and data are not accessible to the second customer and first customer's actions on the first environment will not affect the environment of the second customer. Also, customers are allowed to pay per use i.e. each customer pays for the environment they use and for the time they use the environment rather than purchasing the environment or the components out right.

[53] A further understanding of embodiments of the present invention will be gained with reference to the diagrams and the descriptions that follow.

[54] Fig. 4 illustrates the steps taken by a customer in the process of purchasing computing as a product according to one embodiment of the present invention. In step 410, a customer selects the desired components if needed. Components are presented virtually to customers. Thus customers are only specifying selections. The system presents provides an abstraction for each component such as CPU, storage, OS, applications software, network switches, network bandwidth. Each abstraction may have a set of properties and compatibility constraints associated with it. For instance, Linux operating system software may run only on Intel x86-compatible CPUs; or that Apache web server software may run only under Windows NT/2000 or Linux operating systems; or that a certain switch has 64 ports which limits the number of computers connected to the network controlled by the switch to at most 64.

[55] Once the customer has selected the components the computing environment can be configured in step 420. For instance, the user may specify that the Apache web server software has to run on Linux web servers; or that the web servers and application server(s) must be connected to the same network; or that two different networks must be connected to

each other by a high bandwidth connection; or that a network must have a gateway to the Internet. Step 420 is optional because the system may present abstractions for pre-configured components. Each customer can save the environment configured as a new abstraction. This allows the system to present the new abstraction as a pre-configured component. For instance, a configuration of an Intel x86-compatible computer running a Windows 98 operating system and a WordPerfect word processing system may be presented as an "Easy Word Processing" component. If this configuration meets a customer's requirements then the customer need not do any further work. Of course, the customer may pick a pre-configured component and use it as part of another environment. This method of configuring, saving and presenting environment configurations saves time and effort for customers as well as providers. Environment can be used again and again without going through the configuration step. Commonly used environment configurations can be provided by service providers and used by large numbers of customers.

[56] In step 430 a customer may schedule a period of time for computing. This allows for the system to reserve the required resources and provide a guarantee to the customer on availability. In step 440 a customer "computes" i.e., uses the environment or programs the environment. The customer can repeat steps 430 and 440 as often as needed with the same environment. Eventually the customer may release the environment unless the customer opts to buy the environments or for a perpetually renewable license. Then the customer gets billed for the environment over the period of usage.

[57] The flow chart in Figure 4 is but one possible embodiment of a sequence a user can go through to purchase computing. For instance, there may be different billing options such as per-use billing, periodic billing, billing in installments, or a combination thereof. Also, the user may choose multiple environments and use them concurrently, or at different scheduled times. For instance, a web site for selling office equipment may choose an environment with large computing power and high bandwidth during office hours and a cheaper environment with limited computing power and bandwidth during off hours to optimize their costs versus response time to their customers in turn.

[58] A further embodiment of the present invention is a system to provide computing as a resource that can be accessed remotely. In this embodiment customer usage can be metered and customers can be billed per usage. Multiple customers can access their resources concurrently and each user gets secure access to their resource and is provided with a Quality of Service guarantee. Figure 5 illustrates a block diagram of a platform for this embodiment based on a system for providing Dynamic Computing Environments illustrated in Figures 2 and 3.

[59] DCE Framework 510 in Fig. 5 is the same as the framework 342 in Fig. 3. The Resource pool 512 includes all the resources such as capacity devices (i.e., CPUs), storage servers, network switches, licenses etc. Customer channel C1 (530) is a connection from a single customer to the framework 510. This channel enables the customer to create a dynamic computing environment and use it i.e., it is a computing channel. Customer channel C2 (520) is a computing channel for concurrent usage by a second customer. Security and isolation for each channel (522, 532) is supported by this platform. Resource monitoring (514) tools are used to monitor the individual resources being used from the resource pool and ensure their quality. For instance, tools may be provided to ensure high CPU utilization in each capacity device, or to ensure quick recovery from network failures. These tools eventually ensure that the service meets the QoS guarantees (524, 534) provided for each customer channel. Metering system 516 measures or meters the usage for each channel. Measurements include CPU usage, amount of storage, network bandwidth, IP Addresses, software licenses among others. Usage measurements coupled with period of usage for a channel will be used to bill each customer channel (526, 536).

[60] Support for resource monitoring is available in framework 510 (alias framework 342 in Fig. 3) through daemons (such as 361, 363, and 365 in Fig. 3). Furthermore the Fault Tolerance and Load Balancer components of the usability tier 207 are also helpful in providing Quality of Service guarantees (524, 534).

[61] Support for security and isolation is available through the Sessions and SSL components of the usability tier 207 in Fig. 2.

[62] Support for usage monitoring is available through the managers in application server 311 in Fig. 3. For instance, configuration manager 343 keeps track of configurations saved by customers. So usage for a configuration can then be measured by usage of the components of the configuration. For example, the usage for the "Easy Word Processing" configuration described above would include the device utilization for an x86-compatible computer, the license usage for Windows 98 operating system and the license usage for WordPerfect software. Similarly, the storage manager keeps track of the amount of storage used per customer. Conversion of the usage costs to bill prices is supported by the Billing component of usability tier 207 in Fig. 2.

[63] Thus the platform illustrated in Fig. 5 provides computing as a remote, pay-per-use resource that is scalable and secure.

[64] Although the present invention has been discussed with respect to specific embodiments, one of ordinary skill in the art will realize that these embodiments are merely illustrative, and not restrictive, of the invention. The scope of the invention is to be determined solely by the appended claims.